round(double d)--返回double型d的近似值（四舍五入），返回bigint型；

round(double d,int n)--返回保留double型d的n位小数double型近似值（四舍五入）；

floor(double d)--返回<=d的最大bigint值；

ceil(double d)--返回>=d的最小bigint 值；

ceiling(double d)--返回>=d的最小bigint 值；

rand() --每行返回一个double型随机数；

rand(int seed) --每行返回一个double型随机数，整数seed是随机因子；

exp(double d)--返回e的 d幂次方，返回double型；

ln(double d)--以自然数为底d的对数，返回double型；

log10(double d)--以10为底d的对数，返回double型；

log2(double d)--以2为底d的对数，返回double型；

log(double base,double d)--以base为底d的对数，返回double型；

pow(double d,double p)--d的p次幂，返回double型；

power(double d,double p)--d的p次幂，返回double型；

sqrt(double d)--d的平方根，返回double型；

bin(bigint i)--二进制i的string类型；

hex(bigint i)--十六进制i的string类型；

hex(string str)--计算十六进制表达的str值；

unhex(string i)--hex(string str)的逆方法；

conv(bigint num,int from_base,int to_base)--将bigint类型的num从from_base进制转换成to_base进制，返回string类型；

conv(string num,int from_base,int to_base)--将string类型的num从from_base进制转换成to_base进制，返回string类型；

abs(double d)--计算double型d 的绝对值，返回double型；

pmod(int i1,int i2)--int型i1对 int型 i2取模，结果为int型；

pmod(double d1,double d2)--double型i1对double型 i2取模，结果为double型；

sin(double d)--返回d的正弦值，结果为double型；

asin(double d)--返回d的反正弦值，结果为double型；

cos(double d)--返回d 的余弦值，结果为double型；

acos(double d)--返回d的反余弦值，结果为double型；

tan(double d)--返回d的正切值，结果为double型；

atan(double d)--返回d的反正切值，结果为double型；

degrees(double d)--将弧度值d转换成角度值，结果为double型；

radians(double d)--将角度值d转换成弧度值，结果为double型；

positive(int i)--等价有效表达式是\+i,返回i，结果为int型；

positive(double d)--等价有效表达式是\+d，返回d,结果为double型；

negative(int i)--等价有效表达式是-i，返回i的负数，结果为int型；

negative(double d)--等价有效表达式是-i，返回d的负数，结果为double型；

sign(double d)--如果d是正数的话，则返回float型1.0，如果d是负数的话，则返回-1.0，否则返回0.0；

e()--数学常熟e,超越数；

PI()--数学常数Pi，圆周率；

hive函数之聚合函数

count(*)--计算总行数，包括null值；

count(expr)--计算expr表达式的值，非null的行数；

count(distinct expr[,expr_.])--计算expr表达式的值排重后非null的行数；

sum(col)--指定行的值的和；

sum(distinct col)--排重后值的和；

avg(col)--指定行的值的平均值；

avg(distinct col)--排重后的值的平均值；

min(col)--指定行的值的最小值；

max(col)--指定行的值的最大值；

variance(col)--返回col 中一组数值的方差；

var_pop(col)---返回col 中一组数值的方差；

var_samp(col)--返回col 中一组数值的样本方差；

stddev_pop(col)--返回col 中一组数值的标准偏差；

stddev_samp(col)--返回col 中一组数值的样本偏差；

covar_pop(col1,col2)--返回一组数值的协方差；

covar_samp(col1,col2)--返回一组数值的样本协方差；

corr(col1,col2)--返回两组数值的相关系数；

percentile(bigint int_expr,p)--int_expr在p（范围是[0,1]）处对应的百分比，其中p是double型；

percentile(bigint int_expr,array(p1[,p2…]))--int_expr在p（范围是[0,1]）处对应的百分比，其中p是double型数组；

percentile(double col,p[,NB])--col在p（范围是[0,1]）处对应的百分比，其中p是double型,NB是用于估计的直方图中的仓库数量（默认10000）；

percentile_approx(double col,array(p1[,p2…])[,NB])--col在p（范围是[0,1]）处对应的百分比，其中p是double型数组,NB是用于估计的直方图中的仓库数量（默认10000）；

histogram_numeric(col,NB)--返回NB数量的直方图仓库数组，返回结果array<struct{'x','y'}>中的值x是中心，y是仓库的高；

collect_set(col)--返回集合col元素排重后的数组；

set hive.map.aggr=true; --通过设置属性hive.map.aggr值为true来提高聚合性能；

hive函数之表生成函数

当时用表生成函数时，hive要求使用别名；

explode(ARRAY array)--返回0到多行结果，每行都对应输入的array数组中的一个元素；

explode(MAP map)--返回0到多行结果，每行对应每个map键-值对，其中一个字段是map键，另一个字段是对应的map值；

explode(ARRAY<TYPE> a)--对于a的每个元素，explode()会生成一行记录包含这个元素；

inline(ARRAY<STRUCT[,STRUCT]>)--将结构体数组提取出来并插入到表中；

json_tuple(STRING jsonStr,p1p2,…,pn)--本函数可以接受多个标签名称，对于输入的json字符串进行处理，这个与get_json_object类似，不过更高效，通过一次调用就可以获得多个键值；

parse_url_tuple(url,partname1,partname2,…,partnameN)--从url中解析出n部分信息，其输入参数是url，以及多个要抽取部分的名称。所有输入的参数类型都是string，部分名称大小写是敏感的，不应该包含空格：
HOST,PATH,QUERY,REF,PROTOCOL,AUTHORITY,FILE,USERINFO,QUERY:<KEY_NAME>；

stact(int n,col1,col2,…,colM)--把M列换成N行，每行有M/N个字段，n为常数；

hive函数之其他内置函数

ascii(string s)--返回字符串s中首个ASCII字符的整数型；

base64(binary bin)--基于二进制值bin转换成基于64位的字符串；

binary(srting s)--将输入的值转换成二进制值；

binary(BINARY b)--将输入的值转换成二进制值；

cast(<expr> as <type>)--将expr转换成type类型的，例如cast('1' as bigint)会将字符串转换成bigint数值类型，如果转换失败，则返回null；

concat(binary s1,binary s2,…)--将二进制字节码按次序拼接成一个字符串；

concat(string s1,string s2,…)--将字符串s1,s2等拼接成一个字符串，例如concat('ab','cd')的结果是 'abcd'；

concat_ws(string separator,string s1,string s2,…)--与concat类似，不过是使用指定的分隔符进行拼接的；

context_ngrams(array<array<string>>,array<string>,int K,int pf)--与ngrams类似，但是从每个外层数组的第二个单词数组来查找前K个字尾；

decode(binary bin,string charset)--使用指定的字符集charset将二进制bin解码成字符串（支持的字符集有：'US_ASCII','IOS-8859-1','UTF-8','UTF-16BE','UTF-16FE','UTF-16'），如果任一项输入参数为null，则结果为null；

encode(string src,string charset)--使用指定的字符集charset将字符串src编码成二进制值（支持的字符集有：'US_ASCII','IOS-8859-1','UTF-8','UTF-16BE','UTF-16FE','UTF-16'），如果任一项输入参数为null，则结果为null；

find_in_set(string s,string commaSeparatedString)--返回在以逗号分隔的字符串中s出现的位置，如果没找到则返回null；

format_number(number x,int d)--将数值x转换成'#,###,###.##'格式字符串，并保留d位小数。如果d为0，那么输出值就没有小数点后面的值；

get_json_object(string json_string,string path)--从给定路径上的json字符串中抽取json对象，并返回这个对象的json字符串形式。如果输入的json是非法的，则返回null；

in--例如test in(val1,val2,…),其表示如果test值等于后面列表中任一值的话，则返回true；

in_file(string s,string filename)--如果文件名为filename的文件中有完整一行数据和字符串s完全匹配的话，则返回true；

instr(string str,string substr)--查找字符串str中子字符串substr第一次出现的位置；

length(string s)--计算字符串s的长度；

locate(string substr,string str[,int pos])--查找字符串str中pos位置后字符串substr第一次出现的位置；

lower(string s)--将字符串中所有字母转换成小写字母；

lcase(string s)--和lower()一样；

lpad(string s,int len,string pad)--从左边开始对字符串s使用字符串pad进行填充，最终达到len长度为止。如果字符串s本身长度比len大的话，那么多余部分会被去除；

ltrim(string s)--将字符串s前面出现的空格全部去除掉；

ngram(array<array<string>>,int N,int K,int pf)--估计文件中前K个字尾。pf是精度系数；

parse_url(string url,string partname[,string key])--从url中抽取指定部分的内容。参数url表示一个url字符串，参数partname表示要抽取的部分名称，其是大小写敏感的，可选的值有：
HOST,PATH,QUERY,REF,PROTOCOL,AUTHORITY,FILE,USERINFO,QUERY:<KEY>；如果partname是QUERY的话，那么还需要指定第三个参数key；

printf(string format,Obj…args)--按照printf风格格式化输出输入的字符串；

regexp_extract(string subject,string regexp_pattern,string index)--抽取字符串subject中符合正则表达式regexp_pattern的第 index个部分的子字符串；

regexp_relpace(string s,string regex,string replacement)--按照java正则表达式regex将字符串s中符合条件的部分替换成replacement所指定的字符串a，如果replacement部分是空的话，那么符合正则的部分将会被去除掉。如
regexp_relpace('hive','[ie]','z')的结果是'hzvz'；

repeat(string s,int n)--重复输入n次字符串s；

reverse(string s)--反转字符串；

rpad(string s,int len,string pad)--从右面开始对字符串s使用字符串pad进行填充，最终达到len长度为止，如果字符串s本身长度比len大的话，那么多余部分将会被去除；

rtrim(string s)--将字符串s右面出现的空格全部去除掉；

sentences(string s,string lang,string locale)--将输入字符串s转换成句子数组，每个句子又由一个单词数组构成，单词lang和locale是可选的，如果没有使用，则使用默认本地化信息；

size(MAP<K.V>)--返回map中元素的个数；

size(ARRAY<T>)--返回数组array的元素个数；

space(int n)--返回n个空格；

split(string s,string pattern)--按照正则表达式pattern分割字符串s,并将分割后的部分以字符串数组的方式返回；

str_to_map(string s,string delim1,string delim2)--将字符串s按照指定分隔符转化成map，第一个参数是输入的字符串，第二个参数是键值对之间的分隔符，第三个参数是键和值之间的分隔符；

substr(string s,string start_index,string length)--对于字符串s，从start位置开始截取length长度的字符串，作为子字符串，例如substr('abcdefgh',3,5)的结果是'cdefg'；

substring(string s,string start_index,string length)--对于字符串s，从start位置开始截取length长度的字符串，作为子字符串，例如substr('abcdefgh',3,5)的结果是'cdefg'；

substr(binary s,string start_index,string length)--对于二进制字节值s，从start位置开始截取length长度的字符串，作为子字符串；

substring(binary s,string start_index,string length)--对于二进制字节值s，从start位置开始截取length长度的字符串，作为子字符串；

trim(string a)--将字符串a前后出现的空格全部去掉；

unbase64(string str)--将基于64位的字符串str转换成二进制值；

upper(string a)--将字符串中所有的字母转换成大写字母；

ucase(string a)--将字符串中所有的字母转换成大写字母；

from_unixtime(bigint unixtime[,string format])--将时间戳秒数转换成UTC时间，并用字符串表示，可以通过format规定的时间格式，指定输出的时间格式；

unix_timestamp()--获取当前本地时区下的当前时间戳，例如：1534132825；

unix_timestamp(string date)--输入的时间字符串格式必须是yyyy-MM-dd HH:mm:ss,如果不符合则返回0，如果符合则将此时间字符串转换成Unix时间戳，例如unix_timestamp('2009-03-20 11:30:01')=1237519801；

unix_timestamp(string date,string pattern)--将指定时间字符串格式转换成Unix时间戳，如果格式不对，则返回0，例如unix_timestamp('2009-03-20','yyyy-MM-dd')=1237478400；

to_date(string timestamp)--返回时间字符串的日期部分，例如：to_date("1970-01-01 00:00:00")="1970-01-01"；

year(string date)--返回时间字符串中的年份并使用int类型表示。例如：year("1970-01-01 00:00:00")="1970"；

month(string date)--返回时间字符串中的月份并使用int类型表示。例如：month("1970-01-01 00:00:00")="1"；

day(string date)--返回时间字符串中的天并使用int类型表示。例如：
day("1970-01-01 00:00:00")="1";

dayofmonth(string date)--返回时间字符串中的天并使用int类型表示。例如：
day("1970-01-01 00:00:00")="1";

hour(string date)--返回时间字符串中的小时并使用int类型表示。例如：
hour("1970-01-01 11:58:59")="11";

minute(string date)--返回时间字符串中的分钟数；

second(string date)--返回时间字符串中的秒数；

weekofyear(string date)--返回时间字符串位于一年中第几个周内，例如：
weekofyear("1970-11-01 11:58:59")="44";

datediff(string enddate,string startdate)--计算开始时间startdate到结束时间enddate相差的天数，例如 datediff('2009-03-01','2009-02-27')=2;

date_add(string startdate,int days)--为开始时间startdate增加days天。例如：date_add('2018-12-31',1)='2009-01-01';

date_sub(string startdate,int days)--从开始时间startdate减去days天。例如date_sub('2008-12-31',1)='2018-12-30';

from_utc_timestamp(TIMESTAMP timestamp,STRING timezone)--如果给定的时间戳并非UTC,则将其转化成指定的时区下的时间戳；

to_utc_timestamp(TIMESTAMP timestamp,STRING timezone)--如果给定的时间戳是指定的时区下的时间戳，则将其转化成UTC下的时间戳；